使用 Node.js 技术，建设灵活高效的企业级 Web 系统

文 / 胡扬帆

笔者作为技术实验室自主开发小组成员，近几年使用 Node.js 相关技术，参与了新华社综合评审系统、神笔大侠闯江湖在线游戏、中国重大外交表述语料库等多个社内创新项目的技术设计和开发工作。本文通过总结 Node.js 技术在应用开发工作中的经验与思路，旨在进一步提高自身能力素质，同时就 Node.js 技术如何在我社技术系统中全面推行，更好地应用于核心业务系统的建设进行了探讨，提出了思路和建议。

1. Node.js 技术特性

Node.js 是一个开源跨平台的服务器端 JavaScript 运行环境。它以 Google Chrome V8 解析引擎为基础，采用类似 Nginx 的以事件循环驱动的异步 I/O 模型提高运行效率，通常认为，异步 I/O 的并发性能是同步 I/O 的近 10 倍。Node.js 的核心优势在于通用、灵活、高性能。

Node.js 高性能的本质，在于它使用了底层库 libuv 与操作系统内核配合处理异步事件，libuv 封装了对各操作系统高效事件轮询函数的调用，如 Linux 的 epoll，Unix 的 kqueue，Windows 的 IOCP，libuv 将 Node.js 中高时延的网络 I/O 和磁盘 I/O 的请求通过系统调用注册给操作系统内核，操作系统内核准备好请求数据后回调 libuv 所注册的用户函数将数据送回用户空间完成 I/O。通过这种单线程非阻塞的异步分层协作机制，既解放了 CPU 的高速运算能力，又避免了多进程或多线程模型容易造成的资源调度与开销问题，使得 Node.js 可以像 Nginx 那样单个线程就能处理成千上万的网络请求。

Node.js 暴露给开发者的编程接口十分简洁，开发者只需以回调函数的方式操作 I/O 返回数据就能实现异步 I/O。就这样 Node.js 把易学易用的 JavaScript 和强大的 Unix 网络编程模型结合在一起，在庞大的开源社区以及活跃的技术委员会的带动下，逐渐从各类技术体系中脱颖而出。

Node.js 的不足在于它单线程的运行方式不能有效利用多核 CPU 进行运算，虽然调用 C++ 扩展模块或 Java Jar 包可以弥补，但在原生语言方面它的效率不如多线程的静态编译型语言，因而 Node.js 通常采用 Master-Worker 的主 - 从多进程模式进行部署，以弥补单线程对 CPU 利用率的不足。

2. Node.js 技术的现状

2. 1 全球生态发展现状

笔者从语言生态、开源社区和 Node.js 官方组织的报告等方面调研了 Node.js 技术的现状，近两年 Node.js 相关技术被越来越多的开发者或企业所接受，在社区生态中呈现出前所未有的发展速度。具体情况见附件部分。

2. 2 行业应用现状

美联社早在 2014 年就用了 Node.js 技术对外提供新闻图片查询的公共接口服务。纽约时报在 2017 年 6 月改版了包括 PC 端和移动端的新闻发布产品线，新版后台以 Node.js 服务提供统一支撑，前端使用 React 技术，整个新闻发布业务应用 Node.js 和 JavaScript 全栈技术，这使他们将多语言分别维护的代码仓库合并成了一份，业务变得更加高效、精简。

阿里巴巴作为国内最早一批研究 Node.js 的互联网企业，自 2015 年开始采用 Node.js 技术分担“双十一”线上大流量以来，已经积累了相当多的经验。Node.js 被运用于阿里内部的分布式基础设施之上，替换了原来使用 PHP 或 Java 开发的应用层的程序，利用其灵活高效和高性能的特点服务于数据 I/O 并发量高，运营和业务相关需求多变的应用诉求。使用 Node.js 进行全栈开发提高了开发效率并使前后端合作边界后移。

在阿里的 Java / C++ 和 Node.js 多语言生态中，静态编译型语言开发的底层系统应对数据库事物 ACID、分布式和计算密集场景，业务相对成熟稳定；以 Node.js 为主的上层应用系统应对灵活多变的业务诉求和数据密集的场景，各取所长。除了阿里，国外的 Paypal、Netflix 等体量比较大的公司也使用 Node.js 来做类似的分工。

2. 3 在 Web 应用层对比 Java 的应用优势

应用层的主要任务是与各种数据源和底层系统使用基于 HTTP 或者 RPC 的 API 交流，加上一定的业务逻辑，对数据做适当的处理后，渲染 HTML 或者拼装 JSON 与客户端沟通。应用层的特点在于业务需求多变，接口并发需求高。

首先，在语言方面，强类型静态语言在需要大量处

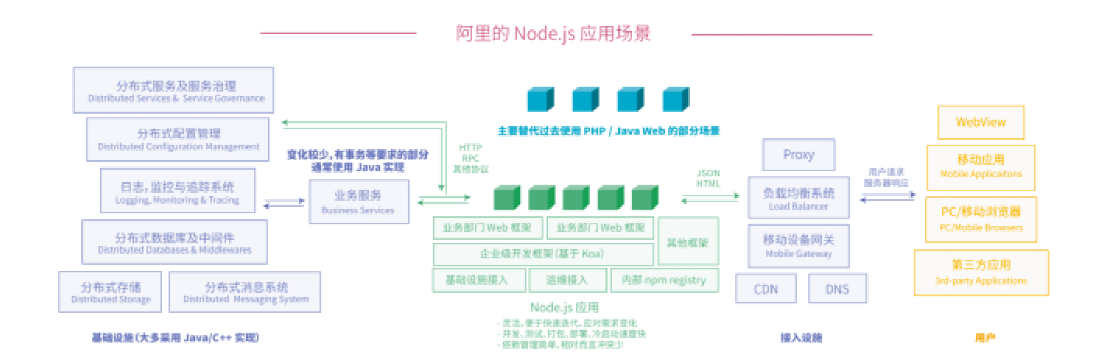


图 1　Node. js 在阿里的应用场景

理 HTML 渲染或 JSON 拼装的场景下，效率先天不如弱类型动态语言。比如当应用需要操作一个 API 返回的 JSON 数据时，Java 需要预先定义好与这个 JSON 对应的 POJO 类（包括成员类型定义和 get/set 方法定义），并进行实例化后才能进行操作，而 JavaScript 可以直接将 JSON 字符串实例化成对象进行操作，当返回 JSON 数据的结构或类型发生变化时，Java 相关代码的维护量将比 JavaScript 多很多。

在框架方面，Java Spring 追求用严格的代码检查和层层的封装规定来保障稳定性，面对应用层 MVC 数据、视图、控制器各层模型都需要灵活定制或扩展情况下，过多的仪式化代码牺牲了开发效率。在没有事务性的复杂要求，或者事务被服务进行封装的情况下，传统框架的稳定性优势难以体现。而 Node.js 从开发、测试、部署、冷启用方面的速度更快，效率更高。

在接口性能方面，以同步阻塞 I/O 为主的多线程 Spring 框架在处理大量网络请求时的并发效率与异步 I/O 不在一个量级，只是目前多进程多服务器的集群部署方式在一定范围内掩盖了其并发性能的劣势。

3. 在实践中逐渐形成 Node.js 自研体系，完成初步探索

从 2014 年以来，笔者所在的开发小组从最初的实践项目《轻易轻应用制作工具》中开发提炼了一套自研的技术开发框架，并应用到近几年的各类项目中。由于当时的早期社区没有类似 Java Spring 的成熟型企业级框架，因此我们进行了大量的 Web 系统相关技术的研究和选型，涉及内容包括登录与鉴权，Cookie 与 Session，数据库 ORM，异常处理，定时任务，模板渲染，日志，Web 安全等方面。逐渐形成了一套相对完善的自有框架，可按需进行配置、集成和扩展。这个过程让我们对开发框架的认识从过去的单方面依赖上升到了可自主掌控的程度，深入理解了框架的作用。

在《新华社综合评审系统》项目中，我们使用有限状态机模型管理投票的状态与状态转换规则，使用 WebSocket 实现端到端的实时双向通信，保证评委端在投票环节之外的所有页面自动化切换，以数据库存储为中心的计算与 I/O 分离策略保证投票状态的持久化和恢复能力，以确保在现场出现任何技术故障时可以做到服务的自动恢复和评委页面的自动恢复。最终该项目以实际表现出来的稳定可靠和快捷易用受到广泛好评。

在《神笔大侠闯江湖微信 H5 游戏》项目中，采用数据库的存储类型优化和稀疏化索引数据的手段提高积分排名接口对实时计算的性能要求，以单进程部署服务进行测试，复杂接口在 5 秒 500 并发的压力下平均响应时间在 2 秒以内，符合性能预期。最终游戏静态资源采用 CDN 分发，接口服务采用了单台虚拟机部署，上线后接口服务快速平稳，应用内存占用平均 300M 左右，没有触及性能天花板。

在后续项目中我们开始实践了整合度更高的全栈开发，例如使用流行的 MVVM 框架 Vue.js 进行前端业务的工程化开发，对接 Node.js 提供的 RESTful 接口服务，对全栈开发与集成的高效性和统一性有实际体会。

凭借 Node.js 技术的灵活特性和动态语言灵活易用的特点，各项开发工作在编码、测试、部署、构建和冷启动等方面效率都很高，使得开发小组在人手不足，任务紧急而艰巨的情况下经常能达到早于预期的进度，并且保障工作任务的完成质量。

4. Node.js 在我社具有广泛的应用前景

随着我社新闻业务向互联网思路转型，整体业务构建于互联网数字化新闻的采集、编辑、发布体系之上，技术特点趋向于数据交互密集，新闻业务模式亦从稳定趋向于灵活创新。其中采集端重在上传效率，编辑端重在协同、检校与流转，发布端重在高并发，稿件全域追溯与流程控制重在各系统服务间的高度整合。数据交互环节多，流程长，涉及数据存取、API 网关、多协议通信、

统一认证、实时消息、流式传输和模板渲染、协同编辑、持续集成等，属于 Node.js 相关全栈技术擅长的应用范围。

对于社内业务稳定的基础组件和基础中间件，计算密集型业务，以及对事务性操作有极高一致性要求的业务应继续保持 Java/C++ 等静态语言的既有优势。其他以功能灵活，数据交互为主的，要求快速开发、高性能的 Web 服务，都可以选择 Node.js 技术进行开发。

软件的组织形式在向着灵活发展，无论从部署形式的 IaaS、CaaS、PaaS、SaaS 到现在的 FaaS（函数即服务，Serverless 架构），从软件组织形式的单一应用到微服务架构，指导思想都是尽量将变化约束到最小的独立单元，以稳定的体系去支撑灵活多变的独立自治的业务单元。因此，对技术选型的要求也从过去的大一统技术框架变成了依据业务特点灵活选择。Node.js 及其 npm 的开源生态践行了单一职责原则的“微服务”的思想，各类技术框架都体现了极简易集成的特点，因此在当前的微服务架构之下，Node.js 可以很好地与不同语言的服务管理框架进行集成接入。

5. 应用 Node.js 技术，更好地进行技术系统的开发

在如今开源社区繁荣生态的支撑下，各类通用性的开发、测试和部署的工具库百花齐放，开箱即用，应用开发人员的核心任务逐渐聚焦于自身业务的实现。通常，我们可以参考工具库提供的应用案例来实现应用功能，可以参考业界高性能架构案例来实现更高性能的服务。但是，业务模型和代码架构只能由我们自己深入理解业务需求后自行设计，因此应用开发的难点在于深度理解业务模型，并对框架和业务代码进行高度抽象以形成高内聚，低耦合的易复用易扩展的结构。

为了更好地将 Node.js 技术应用于系统建设，应努力做到工程的结构清晰、功能灵活、部署稳定、代码便于扩展与维护，主要体现在以下四个方面：

从业务层面，要联系实际抓业务重点，分清主次，了解使用场景，掌控工程实现细节和粒度，广泛了解各种技术框架的特点并做到合理选型，整合人力资源，技术资源，时间资源做切实可行的开发计划。

从技术层面，要有在开源库中找“轮子”，选“轮子”和集成“轮子”的能力；灵活利用开发框架提供的结构性优势；明确业务代码与框架的职责边界；梳理业务代码的层次并实现通用代码的复用；在保证工具、架构和开发流程稳定的情况下实现功能和性能的扩展；配置统一化标准化；使用云计算或容器进行应用部署；实现持续交付。

从运行层面，要考察系统各项 APM 性能监测指标，包括运行进程、线程的稳定性，接口性能，日志追踪，内存占用以及 Web 安全。

从迭代层面，要根据系统运行指标或者新业务诉求对系统进行持续升级改进，当功能依赖开源工具时，按需做到随工具的升级而更新。

综合 Node.js 相关技术以及 JavaScript 技术通用、灵活、高效的特点，全球性的高速发展态势，以及本社新媒体创新业务对技术系统提出的高效性、灵活性、可控性要求，Node.js 相关技术的应用潜力巨大。

附件：Node.js 全球生态现状调研

截止 2018 年 2 月，根据 modulecounts.com 发布的各语言包管理器公共模块数量统计排名，Node.js 包管理工具 npm 诞生 3 年后在 2015 年超越了 Java Maven 成为第一，当前 npm 的模块数量已经数倍于其后的 Java 和 PHP。

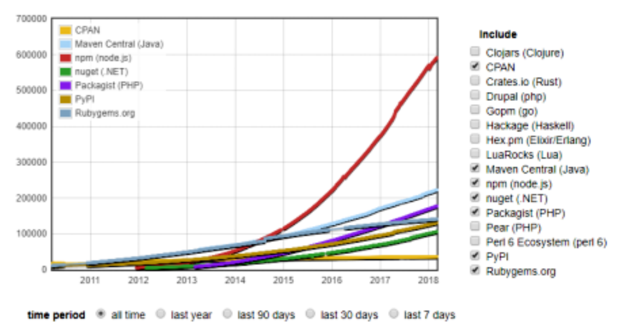


图 1　公共模块数量统计排名（来自 modulecounts.com）

根据 GitHub.com 发布的 Octoverse 报告，JavaScript 成为 2017 GitHub 上最受欢迎的编程语言，JavaScript 技术栈的项目数量超过 Python 或 Java 的两倍。

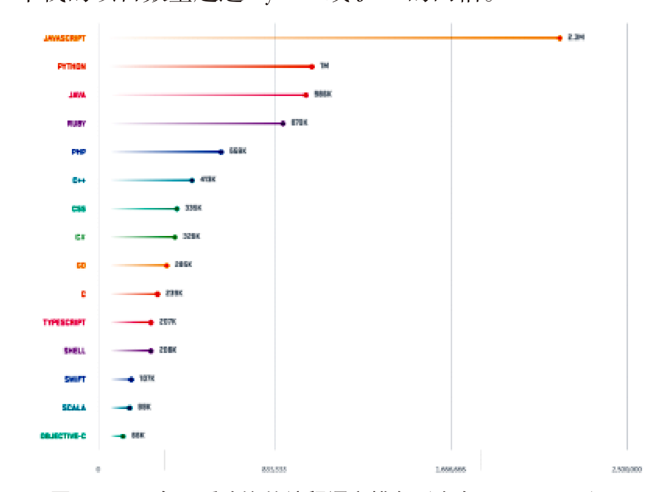


图 2　2017 年最受欢迎的编程语言排名（来自 github.com）

2017 年 Stack Overflow 年度开发者报告中最受开发者

欢迎的技术框架部分 Node.js 排名第一。

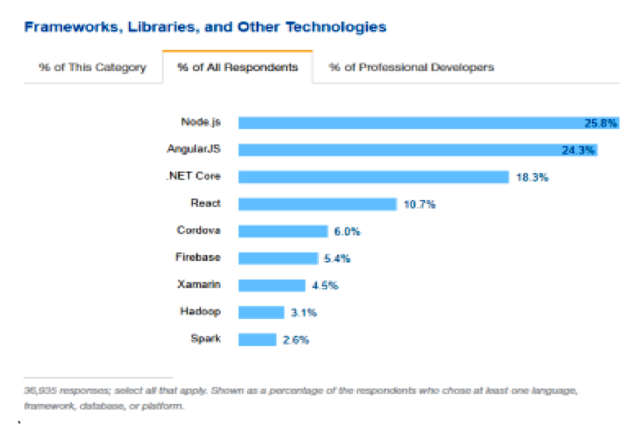


图 3　最受开发者欢迎的技术框架（来自 Stack Overflow）

Node.js 基金会发布的 Node.js 2017 用户调查报告显示，Node.js 覆盖了前端、后端以及全栈应用的开发，最多涉及 Node.js 的工具和技术包括数据存储，前端框架，后端框架，负载均衡，云和容器，持续集成工具，Node.js 在 Web 应用开发中最受欢迎，占比达 84%。

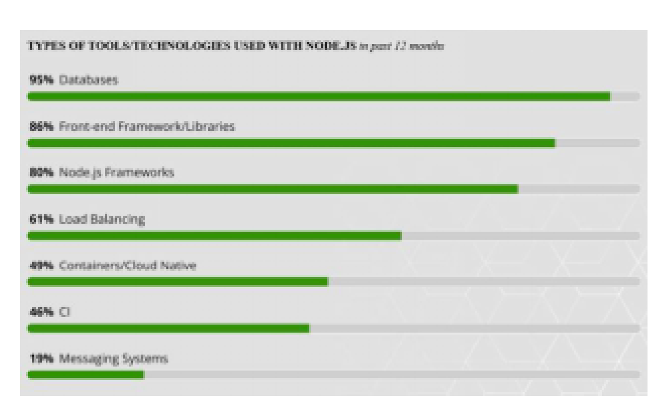


图 4　涉及 Node.js 开发的工具技术

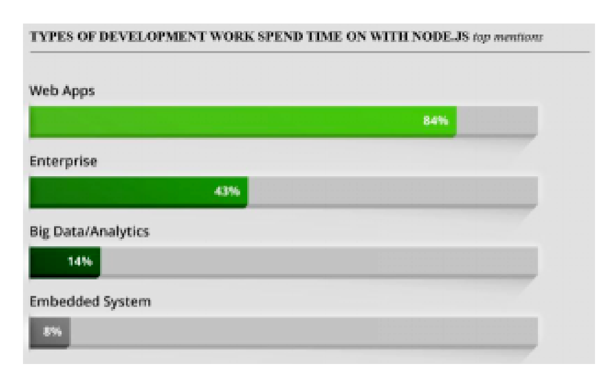


图 5　Node.js 开发场景类型

Node.js 继续保持着前所未有的发展速度，如今它已经成为一个通用的开发框架，被广泛地应用于各种数据数字化的应用程序中。

（作者单位：新华社通信技术局）